

MELHORANDO A EXATIDÃO DE CÁLCULO DA TRANSFORMADA WAVELET DE DAUBECHIES

VINÍCIUS SANTOS¹; RENATA REISER²; MAURÍCIO PILLA²; ALICE
KOZAKEVICIUS³

^{1,2} Universidade Federal de Pelotas – {vrdsantos, reiser, pilla}@inf.ufpel.edu.br

³ Universidade Federal de Santa Maria – alicek@ufsm.br

1. INTRODUÇÃO

Este trabalho considera a Matemática Intervalar (IM) e propõe uma solução baseada na aritmética de (MOORE, 1979), visando melhor conhecimento e controle dos erros nos dados e processos associados à computação científica.

A motivação para uso de técnicas intervalares integradas às Transformadas Discretas *Wavelets* (TDW) é prover resultados confiáveis e automaticamente validados às aplicações que necessitam dessas transformações, cujas áreas são as mais variadas, destacando-se neste contexto o processamento de sinais e de imagens, veja OM; BISWAS (2012) e KUMAR et al. (2012).

Assim, a principal meta deste trabalho consiste no **desenvolvimento de extensões intervalares de TDWs**, sendo neste trabalho abordada a **Transformada Wavelet de Daubechies (TWD)**. Nesta etapa, propõe-se uma implementação da TWD utilizando C-XSC, consistindo num estudo de caso para a implementação da biblioteca Int-DWTs. Este trabalho, insere-se no Projeto ApliFlexD-GM: Consolidando Aplicações Flexíveis Fundamentadas na Lógica Fuzzy, com Suporte na Computação Quântica e Simulação via Ambiente D-GM.

A metodologia de desenvolvimento do trabalho consiste no estudo e na análise dos algoritmos que viabilizam a computação da TWD, reportada em (NIELSEN, 1998). A partir dos algoritmos originais, desenvolveu-se uma biblioteca contendo extensões intervalares, denominada Int-DaWT, sendo introduzida aqui também.

Durante o estudo das formulações originais para a transformada em (NIELSEN, 1998), percebeu-se a possibilidade de otimizá-las, executando simplificações algébricas a fim de se eliminar o cálculo de valores irracionais, originalmente responsáveis pelas normalizações das transformações e considerados em cada iteração de seus algoritmos usuais. Através dessas simplificações, obteve-se um ganho de precisão na formulação intervalar obtida.

Este artigo está organizado da seguinte forma: na Seção 2, são discutidas as otimizações realizadas neste trabalho. Os testes e resultados são apresentados na Seção 3. Finalmente, na Seção 4 as conclusões e os trabalhos futuros são apresentados.

2. OTIMIZAÇÕES

Considerando a utilização de filtros normalizados, a TWD é executada por meio de divisões utilizando $\sqrt{2}$, além de multiplicações por $\sqrt{3}$. Como estes são números irracionais, e por tanto não computáveis, cada nível de decomposição ou composição adiciona um certo grau de erro nos dados computados por conta de truncamentos e arredondamentos. Este erro gerado a cada iteração é então propagado através dos vários níveis de transformação até o fim do processo.

¹ Bolsista de iniciação científica (PIBITI/FAPERGS). Projeto parcialmente financiado pelo Projetos: 309533/2013-9 (PQ-2 CNPq), FAPERGS PQG-1873-2551/13.0, 309533/2013-9 (PqG FAPERGS) e 448766/2014-0 (MCTI/CNPQ/Universal 14/2014 - Faixa B).

Transformações normalizadas preservam a energia dos conjuntos de coeficientes obtidos através do processo. De acordo com o estudo de NIELSEN (1998) e a estratégia utilizada em (DOS SANTOS, 2015) para a Transformada Wavelet de Haar (TWH), que por sua vez é um caso particular da TWD, é possível realizar tanto transformações normalizadas como também não normalizadas.

A solução proposta neste trabalho para evitar erros de cálculo na TWD está em realizar simplificações algébricas com a finalidade de eliminar a computação de valores irracionais, $2^{j/2}$, sempre que possível, reduzindo assim o erro envolvido durante a transformação.

A idéia básica para as simplificações algébricas está baseada na execução da TWD utilizando filtros não normalizados, e realizar ao final desta um passo de normalização dos coeficientes resultantes da transformação não normalizada, como exemplificado na Figura 1. Ao transferir o passo de normalização para o fim da transformação, o fator $\sqrt{2}$ é aplicado somente uma única vez, diminuindo assim o erro de cálculo envolvido.

$$[9 \quad 7 \quad 3 \quad 5] \rightarrow [6 \quad 2 \quad 1 \quad -1] \rightarrow [6 \quad 2 \quad \frac{1}{\sqrt{2}} \quad \frac{-1}{\sqrt{2}}]$$

Entrada \rightarrow **Decomposição** \rightarrow **Normalização**

Figura 1: Base de otimização proposta.

Como visto em (DOS SANTOS, 2015), a execução de TWH não normalizada é realizada ao substituir o divisor $\sqrt{2}$ presente nos filtros pelo valor inteiro 2 . Porém a TWD opera utilizando filtros mais complexos que a TWH e por isso necessita de outro conjunto de simplificações, como o apresentado em Eq.(1).

$$\begin{aligned} h' &= a \left(\frac{1 + \sqrt{3}}{4\sqrt{2}} \right) + b \left(\frac{3 + \sqrt{3}}{4\sqrt{2}} \right) + c \left(\frac{3 - \sqrt{3}}{4\sqrt{2}} \right) + d \left(\frac{1 - \sqrt{3}}{4\sqrt{2}} \right) \\ &= \frac{a(1 + \sqrt{3}) + b(3 + \sqrt{3}) + c(3 - \sqrt{3}) + d(1 - \sqrt{3})}{4\sqrt{2}} \\ &= \frac{a + d + 3(b + c) + \sqrt{3}(a + b - c - d)}{4\sqrt{2}} \end{aligned} \quad (1)$$

A Eq.(1) demonstra as etapas de manipulação algébrica utilizadas para evidenciar os valores irracionais presentes no conjunto de filtros h , reduzindo assim o erro produzido durante as várias iterações do programa. A mesma estratégia de simplificação algébrica é utilizada para se adquirir os filtros g' , ih' e ig' , como mostram as Eqs.(2), (3) e (4).

$$g' = \frac{a - d + 3(c - b) + \sqrt{3}(b + c - a - d)}{4\sqrt{2}} \quad (2)$$

$$ih' = \frac{c + d + 3(a + b) + \sqrt{3}(b + c - a - d)}{4\sqrt{2}} \quad (3)$$

$$ig' = \frac{a - b + 3(c - d) + \sqrt{3}(c + d - a - b)}{4\sqrt{2}} \quad (4)$$

Durante o estudo da formulação original da TWD 2D (NIELSEN, 1998), padrões de normalização de coeficientes foram identificados. Estes padrões foram analisados no desenvolvimento das otimizações para o algoritmo TWD 2D Padrão, apresentados na Figura 2(a). O algoritmo desenvolvido para executar o procedimento em questão utiliza de passos intermediários de normalização ao final da decomposição completa das linhas e após todas as colunas.

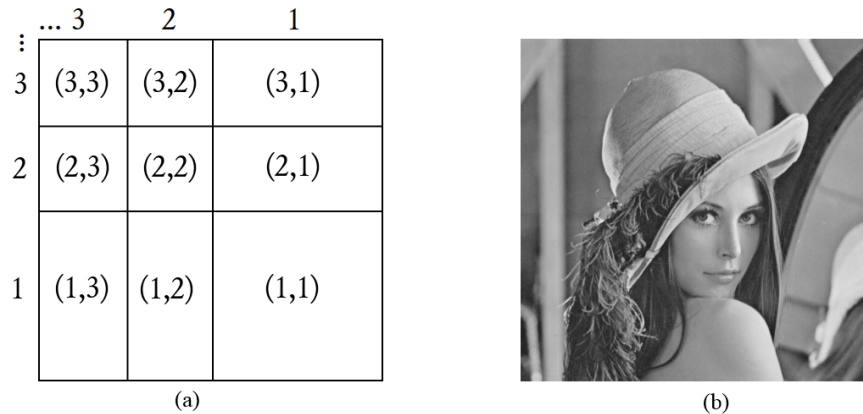


Figura 2: (a) fatores de normalização (j', j'') , (b) parâmetro para testes.

3. RESULTADOS

A medição de tempo é realizada ao executar cada teste 30 vezes. As médias e desvios padrões obtidos destas execuções são apresentadas nos gráficos a seguir. As medições de qualidade dos resultados são realizadas utilizando as métricas *Euclidean Distance (EUC)*, *Mean Square Error (MSE)* e *Peak-to-Noise Ratio (PSNR)*.

Todos os testes foram executados num computador Intel Core i7 950 @ 3.07GHz, 6GB RAM DDR3 @ 1066MHz, Windows 10 OS, utilizando o compilador Microsoft Visual C++ Compiler para o Visual Studio 2013 sobre a arquitetura x64 em modo *Release*. Os dados utilizados nos testes são provenientes da imagem presente na Figura 2(b), com resolução de 4096x4096 pixels.

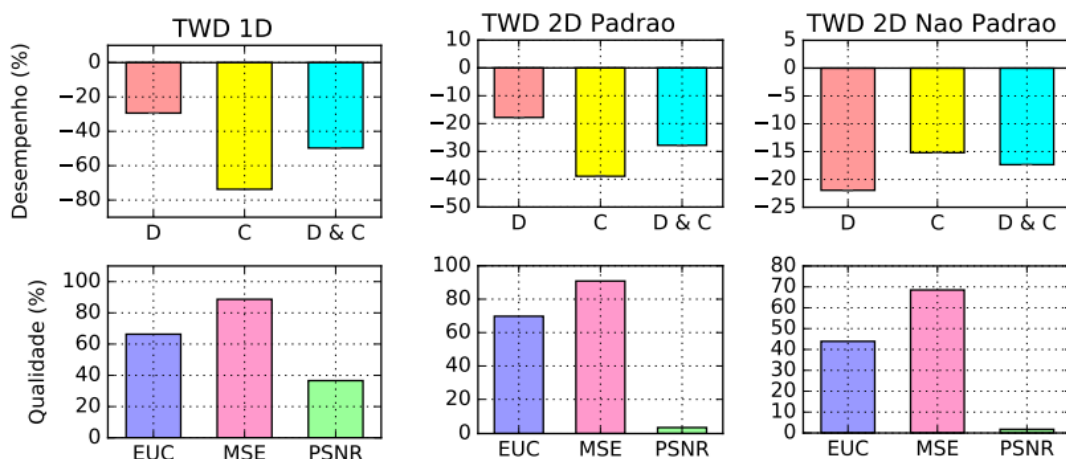


Figura 3: Desempenho e qualidade de resultados da TWD 1D, 2D Padrão e 2D Não Padrão desenvolvidas para as etapas de Decomposição (D), Composição (C) e ambas as etapas (D & C).

Os resultados apresentados na Figura 3 descrevem o desempenho das operações de decomposição e composição com respeito à TWD unidimensional e também suas variações bidimensionais. A TWD 1D apresentou uma perda de

desempenho 26.2% durante a decomposição, 78.3% na composição e 53.6% ao combinar as duas operações. Os tempos das execuções são de 69,0551ms, 88,8472ms e 157,169ms, enquanto que os seus desvios padrões são de 1,55045ms, 0.474684ms e 1,47963ms, respectivamente. As métricas *EUC*, *MSE* e *PSNR* apresentam ganhos de 70.8%, 91.5% e 18.2%, respectivamente. A TWD 2D Padrão apresentou uma perdas de desempenho 17,2% durante a decomposição, 39,1% na composição e 28,5% ao combinar as duas operações. Os tempos das execuções são de 1,0524s, 1,23855s e 2,295s, enquanto que os seus desvios padrões são de 2,34561ms, 2,01274ms e 7,64587ms. As métricas *EUC*, *MSE* e *PSNR* apresentam ganhos de 70.3%, 91.2% e 3.6%. A TWD 2D Não Padrão apresentou uma perdas de desempenho 21,4% durante a decomposição, 15,1% na composição e 17,6% ao combinar as duas operações. Os tempos das execuções são de 1,27712s, 1,79156s e 3,06784s, enquanto que os seus desvios padrões são de 3,8811ms, 4,07062ms e 8,20209ms. As métricas *EUC*, *MSE* e *PSNR* apresentam ganhos de 45%, 69.8% e 1.8%.

4. CONCLUSÃO

Este trabalho apresentou a implementação de otimizações sobre o algoritmo da TWD obtendo bons resultados quanto à análise do erro. As mesmas otimizações viabilizaram o aumento na exatidão dos cálculos, e como consequência também aumentaram no tempo de execução dos mesmos cálculos.

Outras otimizações serão estudadas para se melhorar a indexação dos dados, auxiliando no acesso dos mesmos e assim diminuindo o tempo de execução da transformação.

Na continuidade, busca-se a extensão da biblioteca para suporte a programação paralela, utilizando a linguagem CUDA para paralelização massiva em GPGPUs (*General-Processing Graphics Processing Units*).

5. REFERÊNCIAS

DOS SANTOS, V.; PILLA, M.; REISER, R.; KOZAKEVICIUS, A. (2015). **Int-HWT: Increasing Performance and Exatitute of 1D and 2D Haar Wavelet Transforms**. Simpósio Sul de Microeletrônica, SIM, Santa Maria.

OM, H.; BISWAS, M. (2012). An improved image denoising method based on wavelet thresholding. **Journal of Signal and Inf. Processing**, v.3, p.109–116.

KUMAR, M.; SUDHANSU, S.; KASABEGOUDAR, V. (2012). Wavelet based texture analysis and classification with linear regration model. **International Journal of Engineering Research and Applications**, v.2, p.1963–1970.

NIELSEN, O. M. (1998). **Wavelets in Scientific Computing**. Dissertação (Doutorado). - Informatics and Mathematical Modelling, Technical University of Denmark.

MOORE, R. (1979). **Methods and Applications of Interval Analysis**. SIAM, Philadelphia.

STOLLNITZ, E.; DeROSE, T.; SaALESIN, D. (1995). **Wavelets for computer graphics: A primer, Part 1**. IEEE Computer Graphics and Applications, 15(3):76–84.